Partial Scan Selection for User-Specified Fault Coverage

Clay Gloster Franc Brglez
CBL (CAD Benchmarking Laboratory), Department of Electrical & Computer Engineering
Box 7911, North Carolina State University, Raleigh, N.C. 27695, U.S.A.
(WWW: http://www.cbl.ncsu.edu/www/)

Abstract – *With current approaches to partial scan, it is difficult, and often impossible, to achieve a specific level of fault coverage without returning to full scan. In this paper, we introduce a new formulation of the minimum scan chain assignment problem and propose an effective covering algorithm and test sequence generator* SCORCH *(*Scan Chain Ordering with Reduced Cover Heuristic*) to solve it.* SCORCH *uses a combinational test generator not only to optimize the scan chain assignment, subject to maintaining a user-specified level of fault coverage, but also as a basis for the test sequence generation. We report experimental results with minimized partial scan assignment and 100% fault coverage for a set of large benchmarks.*

I. Introduction

A test strategy based on full scan design can typically achieve 100% fault coverage at an acceptable cost of test generation. The drawback may be the cost in reduced performance, increased area and longer test application time. A partial scan design approach considers only a subset of flip-flops for inclusion in the scan chain and can thus reduce these drawbacks.

In general, approaches to partial scan rely on a combination of one or more measures that are based on testability analysis, circuit topology, in particular as it relates to cycles in the S-graph, and test generation. Recent publications that cover much of this work include [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], and [13].

The approach introduced in this paper uses a sequential fault simulator, [14], [15], and user-supplied or random test sequences to cover as many faults as possible *before* considering the scan chain assignment. Subsequently, we apply a combinational test generator, [15], [16], to generate tests for the remaining set of *hard faults*. These tests provide a formal basis for minimized scan-chain assignments as well as for test sequence generation. Unlike earlier methods, this approach offers flexibility to maintain a user-specified level of fault coverage. Experimental results in this paper have been devised to provide an analytical basis in order to improve comparisons with alternative partial scan approaches. In particular, a guarded comparison is possible with some of the results reported in [5], [17], [18], [19], [20], and [21].

Our experimental results are based on some existing and some new benchmarks [22]. We have chosen this benchmark set for two reasons: (1) all circuits are 100% testable under full scan, (2) none of the benchmarks can be fully tested by applying a sequence of random patterns for 32,768 clock cycles. A more detailed characterization of these benchmarks is given in Table II. An interesting performance graph emerges upon submitting the same netlists to a state-of-the-art sequential test generator such as HITEC [23] and a sequential fault simulator SIFT [14], [15] and is shown in Figure 1. The fault simulator always starts from an unknown state while sequences of random patterns are applied to each circuit for 32,768 clock cycles. Four of the circuits marked with a * (s967*, s991*, s1269*, s1512*), include a reset gate to some of the flip-flops; without them, these circuits are uninitializable. The results of this experiment show that for the circuits where a sequential test generator does find test sequences with high fault coverage, application of random patterns to the fault simulator is just as effective– and much less costly. A circuit such as s6669 presents a real challenge to sequential test generation. After 80,000 CPU seconds spent on sequential test generation, fault coverage is still less than 1%, while sequential fault simulation achieved a fault coverage of 99.87% within 2,611 CPU seconds. Using SCORCH, the combined cost of sequential fault simulation, combinational test generation, partial scan assignment and partial scan test generation for the most difficult circuit in this benchmark set (s3330) requires less than 8000 CPU seconds while achieving 100% fault coverage under partial scan.

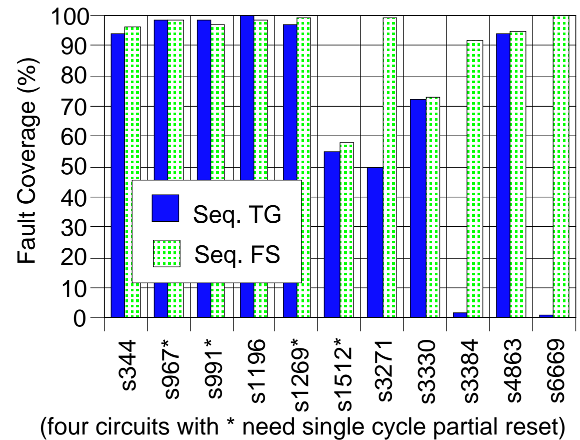

Fig. 1. Fault Coverages of Original Non-Scan Circuits.

II. Problem Formulation for Partial Scan

Consider a circuit with n flip-flops and n decision variables $\mathbf{c} = [c_1, c_2, c_3, \ldots, c_n]$, $\mathbf{c} \in B^n$. A 0-1 assignment to $\mathbf{c}$ determines which flip-flops are in the scan chain. Let FC($\mathbf{c}$) be the fault coverage of this circuit that exceeds the

This work was supported in part by the National Science Foundation under grant MIP-9410793 and by contracts from the Semiconductor Research Corporation (94-DJ-553), SEMATECH (94-DJ-800), and ARPA/ARO (P-33616-EL/DAAH04-94-G-2080).

user-prescribed fault coverage threshold FC_t. The scan chain assignment problem is then formulated as follows:

$$\min_{\mathbf{c} \in B^n} ||\mathbf{c}|| \quad \textit{such that} \quad FC(\mathbf{c}) \geq FC_t \qquad (1)$$

The number of solutions to this problem grows exponentially with the number of flip-flops. An exhaustive enumeration procedure is impractical even for circuits with 20 flip-flops. The circuit example under consideration, s3330 described in more detail in Section V, has a total of 132 flip-flops. As illustrated in Figure 2, the *quality of partial scan solutions* for this circuit can vary over a wide range. By plotting the percentage of flip flops in the scan chain versus the fault coverage for a circuit, we expose the quality, not only of a single solution, but for a whole set of possible solutions. There are two anchor points in Figure 2:

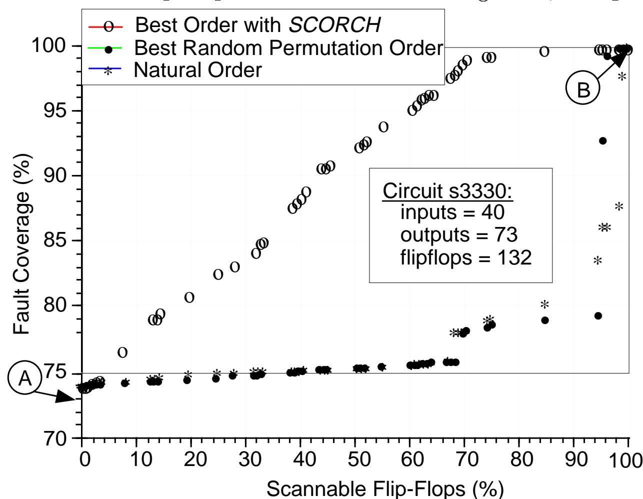

Fig. 2. Fault Coverage versus Scan Chain Length (s3330).

- **A:** 0% scan and the *best initial* fault coverage $FC_{initial} = 73.38\%$.
- **B:** 100% scan and the *best final* fault coverage $FC_{final} = 100.00\%$.

The three curves shown in Figure 2 can be considered as a special case of three solutions based on a given ordering of flip-flops. We can start either at the point A or at the point B. Starting at point B, and removing from the scan chain one flip-flop at a time, we get three monotonically decreasing curves. The initial order of flip-flops is depicted as the *Natural Order*, an order selected after a number of random permutations of the order is termed as the *Best Random Permutation Order*, and the *Best Order with SCORCH* depicts an order that resulted by applying the algorithm SCORCH described in this paper. For fair comparison with the algorithm, the *Best Random Permutation Order* has been executed for $n(n-1)/2$ random permutations of the order, the order of complexity of the *SCORCH* algorithm.

While there are no guarantees that the *Best Order with SCORCH* is really the best, the space of solutions to this problem is clearly very wide and open. For a solution that claims 99% fault coverage or better, one can vary the percentage of scannable flip-flops from 70% to 100%, and if the fault coverage threshold is set at 90%, no more than 43% of the flip-flops need to be in the scan chain. *Therefore, for meaningful comparison of partial scan approaches, all circuits should be compared at the same level of fault coverage.*

III. Notation and Definitions

We consider the model of a synchronous sequential circuit where all flip-flops are controlled by a *single clock*. Inputs and outputs are represented with the binary vectors $\boldsymbol{x}$ and $\boldsymbol{Z}$, respectively. Present state and next state vectors, associated with n flip-flops are $\boldsymbol{y}$ and $\boldsymbol{Y}$, respectively. Vector $\mathbf{c} \in B^n$ represents decision variables that determine flip-flops in the scan chain.

Initially, the sequential circuit represents a single *Object Machine* (*OM*). As shown in Figure 3, a *Partial Scan Test Machine* (*TM*) shares a subset of flip-flops with the *OM*. There are a total of $n_s + n_i < n$ flip-flops in the proposed *TM*: n_s flip-flops form the MUX-based scan-chain, n_i flip-flops are controlled by the initializing gate and an additional primary input that can initialize these flip-flops *in a single clock cycle*. Flip-flops in the *TM* are either in the scan chain or single-cycle initializable, but not both. By maintaining the initializing gates as part of the *TM*, we have the advantage of two distinct test strategies: one that targets the fault set $\mathcal{F}_{OM}$ in *OM*, and one that targets the fault set $\mathcal{F}_{TM}$ in *TM*. In contrast to earlier *TM* implementations that require gated or multiple clocks, the implementation proposed in Figure 3 is completely synchronous with the *OM*, i.e. it requires *a single ungated clock*. Issues of gated versus a single ungated clock for partial scan have been addressed only recently [20].

Given the notation for a step function ($u(x) = 1$ for $x > 0$ and $u(x) = 0$ otherwise), the size of the set of all equivalence-collapsed faults in the *TM* as shown in Figure 3 can be determined using a simple formula:

$$|\mathcal{F}_{TM}| = 9n_s u(n_s) + 5u(n_s) + 3n_i u(n_i) + 2u(n_i) \qquad (2)$$

where n_s and n_i are as described above. We partition the set of all faults in *OM*, $\mathcal{F}_{OM}$, into two subsets: $\mathcal{F}_{initial}$ and $\mathcal{F}_{hard}$. We assume that tests to cover *all* faults in $\mathcal{F}_{initial}$ have been generated, *with 0% scan*, upon applying a test sequence based on random, functional, or deterministic test generation. We denote $f_i \in \mathcal{F}_{hard}$ as the target fault that may require a partial scan assignment in order to generate a test sequence that covers f_i. The size of this set is $m = |\mathcal{F}_{hard}|$.

To illustrate proposed concepts, consider a small *OM* with 9 flip-flops and 6 hard faults that could not be detected by a *sequential* test generator: i.e. $n = 9$, $|\mathcal{F}_{hard}| = 6$. Patterns, called *excitation vectors*, that detect this set with a *combinational* test generator are shown in Table IA. As can be seen, several inputs and flip-flops can be assigned a *don't care value of* '$-$' to excite as well as to propagate the faults. Flip-flops and outputs to which the faults did propagate are designated with D and $\overline{D}$. If a fault, such as f_6, propagates directly to a primary output, then all next state entries in the Table are '$-$'.

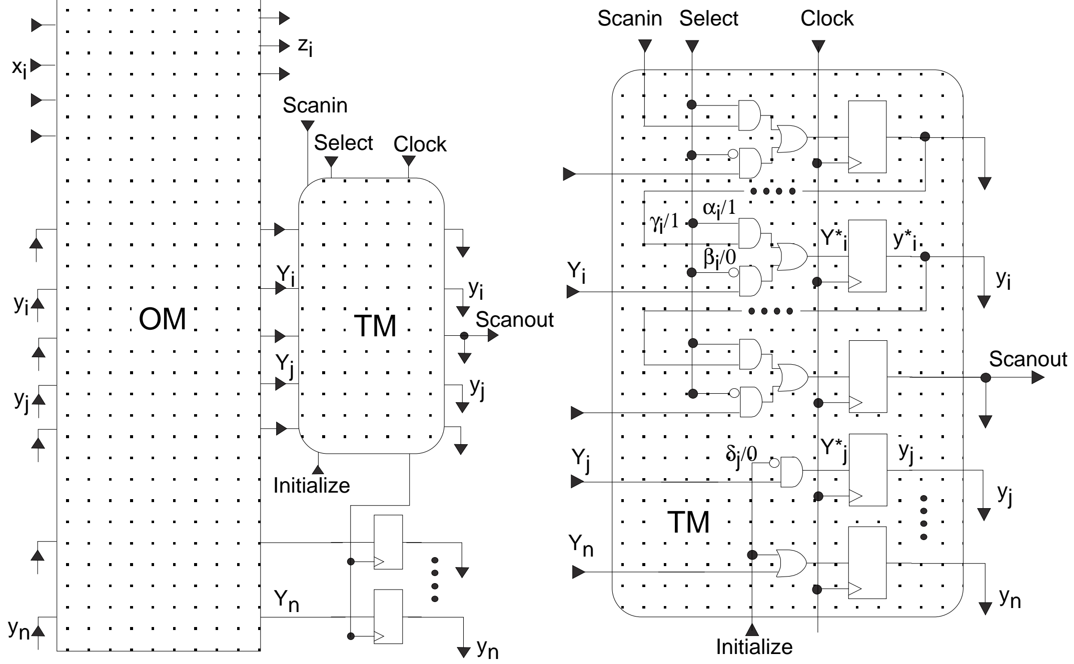

Fig. 3. Single Clock Partial Scan Test Machine Model.

These excitation vectors have useful information on the detectability of faults in $\mathcal{F}_{hard}$. For example, looking at the test pattern generated for f_2 we find that the effects of the fault will propagate to flip-flops 1, 4 and 6 when the pattern *100* is applied to the primary inputs and flip-flops 1, 3, and 9 are assigned a *1*. The fault f_6 propagates directly to primary output, z_1, when *111* is applied to the primary inputs and flip-flop 8 has been set to *1*.

Fault Excitation Matrix, $\boldsymbol{M}^e$, relates variables in the *present state* vector $\boldsymbol{y}$ to a specific fault f_i. The elements of this matrix are s^e_{ij}. We can extract this $m \times n$ matrix from *excitation vectors*, e.g. Table IA. With each fault f_i, we associate a fault excitation state S^e_i and an index set E_i that points to all assigned values of *0* and *1* in S^e_i.

Excitation Status. We generate equations that determine the excitation status of each fault by forming a conjunction of unate Boolean variables that correspond to assigned values in the matrix $\boldsymbol{M}^e$:

$$Ef_i(\mathbf{c}) = \bigwedge_{j \in E_i} c_j \tag{3}$$

For the fault f_1, we can write $Ef_1(\mathbf{c}) = c_2\, c_8$. By including flip-flops 2 and 8 in the chain, we excite f_1 simply by shifting in a *1* to flip-flop 2 and *0* to flip-flop 8.

Fault Capture Matrix, $\boldsymbol{M}^c$, relates variables in the *next state* vector $\mathbf{Y}$ to a specific fault f_i. The elements of this matrix are s^c_{ij}. We can extract this $m \times n$ matrix as a next state response to the *excitation vectors*, shown in Table IA. With each fault f_i, we associate a fault capture state S^c_i and an index set C_i that points to all assigned values of D or $\overline{D}$. We say that a fault f_i is captured in flip-flop j, if $j \in C_i$. If fault f_i propagates to any primary output, we designate all entries in S^c_i as *empty*.

Fault Propagation Matrix, $\boldsymbol{M}^p$, records variables in the *present state* vector $\boldsymbol{y}$ that must be assigned in order to propagate a fault captured in the flip-flop k (denoted as D or $\overline{D}$ on the diagonal of the matrix) to a primary output. The elements of this matrix are s^p_{kj}. We can extract this $n \times n$ matrix from *propagation vectors*, shown in Table IB. These vectors can also be generated with a combinational test generator. With each flip-flop k, we associate a fault propagation state S^p_k and an index set P_k that points to all assigned values of *0* and *1* in S^p_k. If a fault captured in flip-flop k *does not* propagate to any primary output, we designate all entries in S^p_k with "$*$". An example of such a case is flip-flop 1 in Table IA.

Propagation Status. For all *non-empty* capture states S^c_i, we generate equations that determine the propagation status of each fault by the bitwise intersection of entries s^c_{ij}, s^p_{kj} in matrices $\boldsymbol{M}^c$ and $\boldsymbol{M}^p$ and their respective states S^c_i and S^p_k.

$$S^c_i \cap S^p_k \quad = \quad \begin{cases} \textit{perfect} \text{ wrt fault } f_i & \textit{if and only if} \\ \forall j \in P_k \;\, s^c_{i,j} \cap s^p_{k,j} \in V & \\ \textit{imperfect} \text{ wrt fault } f_i & \textit{otherwise} \end{cases} \tag{4}$$

where V is the set:

$$\begin{aligned} V = \quad & \{(0,0),(1,1),(-,-),(D\,,D\,), \\ & (\overline{D}\,,\overline{D}\,),(0,-),(1,-),(D\,,-), \\ & (\overline{D}\,,-),(D\,,\overline{D}\,),(\overline{D}\,,D\,),(D\,,1),(\overline{D}\,,0)\} \end{aligned} \tag{5}$$

Note that for the simple example, $S^c_5 \cap S^p_8$ is perfect since $S^c_5 \cap S^p_8 = \{(-,-),(0,0),(1,-),(\overline{D},-),(\overline{D},\overline{D})\}$, implying that the fault f_5 will always, i.e. *unconditionally*, propagate to a primary output. However, $S^c_3 \cap S^p_8$ is imperfect: $S^c_3 \cap S^p_8 = \{(1,-),(0,-),(-,0),(0,0),(D,\overline{D}),(-,-)\}$, since $(-,0) \notin V$. Hence, the fault f_3 will not propagate

TABLE I

Fault Excitation, Capture, and Propagation Matrices.

(A) EXCITATION VECTORS - TEST PATTERNS FOR UNDETECTED FAULTS IN THE OM.

Fault	x_1	x_2	x_3	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	z_1	z_2
f_1	-	-	0	-	1	-	-	-	-	-	0	-	-	0	-	$\overline{D}$	1	-	-	$\overline{D}$	-	0	-
f_2	1	0	0	1	-	1	-	-	-	-	-	1	D	1	-	$\overline{D}$	0	D	1	1	-	1	1
f_3	0	1	-	0	1	0	-	-	-	-	-	-	1	0	-	0	1	0	0	D	-	-	1
f_4	0	0	0	-	-	-	-	-	-	-	0	-	D	-	-	-	1	1	1	-	-	1	1
f_5	1	-	1	-	-	-	-	-	-	-	-	0	-	-	0	0	1	-	$\overline{D}$	$\overline{D}$	-	0	1
f_6	1	1	1	-	-	-	-	-	-	-	1	-	-	-	-	-	-	-	-	-	-	$\overline{D}$	1

(B) PROPAGATION VECTORS - TEST PATTERNS FOR ALL FLIP-FLOP OUTPUT FAULTS.

FF	x_1	x_2	x_3	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	z_1	z_2
1	0	1	0	D	*	*	*	*	*	*	*	*	D	-	1	$\overline{D}$	0	D	-	0	0	0	-
2	1	0	0	0	$\overline{D}$	-	-	-	-	-	1	-	-	-	-	1	1	0	-	-	-	$\overline{D}$	1
3	0	1	1	*	*	$\overline{D}$	*	*	*	*	*	*	0	0	0	0	1	-	-	D	-	0	1
4	1	1	-	1	-	-	D	-	-	-	0	0	-	0	0	1	0	-	-	1	1	-	D
5	1	1	-	1	0	0	1	D	0	1	1	1	$\overline{D}$	1	1	-	-	-	-	-	-	1	0
6	1	-	-	-	-	-	-	-	$\overline{D}$	-	-	-	1	0	-	-	-	1	-	1	-	$\overline{D}$	1
7	-	0	0	1	-	0	-	1	-	$\overline{D}$	0	-	D	1	1	-	-	-	-	-	-	0	D
8	1	-	0	-	-	0	0	-	-	-	$\overline{D}$	-	-	-	-	1	1	1	1	-	-	D	-
9	1	1	-	-	-	-	-	1	1	-	-	$\overline{D}$	0	-	1	-	-	-	-	D	-	$\overline{D}$	1

to a primary output unless flip-flop 8 is included in the scan chain.

Boolean equations that characterize *propagation status* for each fault are thus:

$$Pf_i(\mathbf{c}) = \begin{cases} 1 & \text{if intersection in (4) is perfect} \\ \bigvee_{j \in C_i} c_j & \textit{otherwise} \end{cases} \quad (6)$$

For the fault f_1, we find $Pf_1(\mathbf{c}) = c_4 + c_8$.

Any fault f_i for which $Pf_i(\mathbf{c}) = 1$ is called *unconditionally observable.* We can show that any flip-flop associated with any unconditionally observable fault f_i is in fact an *implicit scan-out pin* and *need not be included in the scan chain* to propagate the effects of fault f_i directly to a primary output. In our example, flip-flop 8 is an implicit scan-out pin for fault f_5 only, whereas flip-flop 6 is an implicit scan-out pin for any fault f_i.

Detectability Status. Detectability of any fault f_i is a conjunction of the excitation status in (3) and the propagation status in (6). Boolean equations that characterize *detectability status* for each fault are thus:

$$Df_i(\mathbf{c}) = Ef_i(\mathbf{c}) \wedge Pf_i(\mathbf{c}) \quad (7)$$

For the fault f_1, we find $Df_1(\mathbf{c}) = c_2\ c_8\ (c_4 + c_8)$.

Fault Coverage of Hard Faults, $FC_{hard}(\mathbf{c})$ is based on detecting any of the faults from the set $\mathcal{F}_{hard}$.

$$FC_{hard}(\mathbf{c}) = \frac{1}{|\mathcal{F}_{hard}|} \sum_{i=1}^{|\mathcal{F}_{hard}|} Df_i \quad (8)$$

Object Machine Fault Coverage $FC_{OM}(\mathbf{c})$ combines fault coverage $FC_{initial}$ that covers the initial fault set $\mathcal{F}_{initial}$ with coverage of faults in the set $\mathcal{F}_{hard}$:

$$FC_{OM}(\mathbf{c}) = FC_{initial} + (1 - FC_{initial}) \times FC_{hard}(\mathbf{c}) \quad (9)$$

IV. Algorithm SCORCH

Unlike other algorithms for partial scan, *SCORCH* not only makes the partial scan chain assignment, but it also generates test sequences that guarantee the user-prescribed level of fault coverage. The *algorithm* consists of three major phases:

(1) Generation of Excitation, Capture and Propagation Matrices. This process is driven by the circuit netlist and the list of undetected faults, $\mathcal{F}_{hard}$. A modified combinational test generator attempts to minimize assigning values to pseudo-primary inputs (flip-flop outputs) and avoids propagating the fault effects to pseudo-primary outputs (flip-flop inputs). This objective, similar to [12], is achieved by setting the depth of the flip-flop outputs to a very large number.

(2) Scan Chain Assignment. Figure 4 presents a version of the *SCORCH* algorithm where we use a greedy *single flip-flop-at-a-time* removal heuristic to solve (1). The alternative greedy heuristic, adding a single flip-flop at a time, produces equivalent results. Clearly, the algorithm has worst case complexity of $\mathbf{O}(n^2)$. In contrast to more traditional heuristics that address 100% covering [24], our approach also considers the partial covering problem.

(3) Partial Scan Test Generation. The test generation algorithm solves the following problem: *Given an assignment of the flip-flops and the order of the {partial} scan chain flip-flops, derive the set of test patterns that will detect all faults.*

```
Program Name: ScanChainSelection(FC_t, c, FC(c))
Input:  FC_t
Output:  c,FC(c)
/* Start with all FFs in the Scan Chain */
c = { 1, 1, 1, ..., 1};
k=||c||;
while FC(c) >=  FC_t  do
   Find c such that:
      ||c|| = k-1 AND
      FC(c) is maximum;
end while
return(c,FC(c));
```

Fig. 4. **SCORCH** - Scan Chain Assignment

```
Program Name: TestGeneration(c,TS)
Input:  c
Output:  TS
TS = ∅;
for each undetected fault f_i  do
    Generate ES_{f_i};
    TS = (TS, ES_{f_i});
    FF_j = FF which captured f_i
    if FF_j is any of the non-scannable
    unconditionally observable FFs then
        Generate PS_{f_i};
        TS = (TS, PS_{f_i});
    end if
end for
```

Fig. 5. **SCORCH** - Test Generation

We compute a test sequence that detects each fault f_i. The test sequence consists of an excitation sequence ES_{f_i} followed by a propagation sequence PS_{f_i}. The excitation sequence is formed via shifting in the required 0-1 values found in the excitation matrix. The propagation sequence is either a single cycle application of a propagation vector that corresponds to an unconditionally observable flip-flop or a simple shift-out sequence. Since the propagation sequence often consists of a simple shift-out sequence, we can overlap $ES_{f_{i+1}}$ with PS_{f_i} to reduce test application time. Figure 5 shows the process of test generation and takes advantage of overlapping to reduce test application costs.

V. Experimental Results

Partial scan assignment and test generation results reported with SCORCH are based on the initial experimental assessment of circuits as described in Section I, Figure 1. Details about fault simulation and the target fault list $\mathcal{F}_{OM}$ are shown in Table II. Only four circuits require an additional input solely for the purpose of initialization; one circuit (s1512) currently requires a global single clock cycle reset, three circuits require only a partial single clock cycle reset on some of the flip-flops. The remaining six circuits use *no* explicit initialization input, in contrast to a number of initialization flip-flops reported in [17]. Our assignment of initialization flip-flops is based on a simple heuristic, distinct from [25], to be described elsewhere.

TABLE II
Benchmark Characteristics

Circuit	Inps	Outs	FFs	Init FFs	$\mathcal{F}_{OM}$ Faults	$\mathcal{F}_{OM}$ $FC_{initial}$ (%)
s344	9	11	15	0	342	96.20
s967	16	23	29	6	1066	98.22
s991	65	17	19	18	910	97.14
s1196	14	14	18	0	1242	98.55
s1269	18	10	37	26	1343	99.70
s1512	29	21	57	57	1357	57.41
s3271	26	14	116	0	3270	99.57
s3330	40	73	132	0	2870	73.38
s3384	43	26	183	0	3380	91.92
s4863	49	16	104	0	4764	95.11
s6669	83	551	239	0	6684	99.87

As shown in Figure 1, sequential fault simulation of random sequences can reduce the size of the target fault set $\mathcal{F}_{hard}$ much more effectively than sequential test generation. According to Tables II and III, faults targeted by SCORCH for 100% coverage range from 0.3 % (4 faults) for circuit s1269 to 26.62% (764) for circuit s3330. Notably, we do achieve exactly the postulated fault coverage of 100% for the *OM* while the percentage of flip-flops that must be scannable ranges from 96.49% (s1512) to 7.53% (s6669). Some, but not all, of the flip-flops that were initially designated as initializing flip-flops are reassigned to be scannable, e.g. s991. To generate test sequences that cover most, if not all, of the postulated faults in the *TM*, we currently re-apply SCORCH with $\mathcal{F}_{TM}$ as a target fault list. Results reported in Table III indicate near 100% or 100% coverage of $\mathcal{F}_{TM}$. A formal procedure to cover *all* faults in *TM* as well will be reported elsewhere.

At present, it is hard to assess merits of any particular partial scan assignment algorithm. For example, even for a small 15 flip-flop circuit (s344), reports from [5], [17], [18], [19], [20], and [21] range from 1 flip-flop to 8 flip-flops in the chain – while the reported fault coverages range from 96.8% to 100%. Similarly, for the circuit s953, 3 - 6 flip-flops are reported to achieve a fault coverage of 50 - 100%. This finding is consistent with our analysis of experimental results, introduced in Section II, Figure 2. Only the largest circuit s6669 can be compared directly to an earlier result elsewhere: we find 100% fault coverage, 72 scannable flip-flops and 2 initializing flip-flops in [17], and 18 scannable flip-flops and 0 initializing flip-flops in Table III. While there are more circuits in [17] that overlap with ones in Table III, non-trivial variations in reported scan as well as in fault coverage (of faults in OM, TM, or both?) make further comparisons difficult.

The cost of using *SCORCH* for both partial scan chain assignment and test generation is relatively low. The scan chain assignment requires evaluation of Boolean equations and is fast. Test sequence construction is even faster, once we have the proper combinational test vectors and the scan chain assignment. For the circuits in the benchmark set, CPU time on a SUN Sparcstation LX can range from

TABLE III
Test Generation Results

Circuit Name	No. of FFs n	Init FFs n_i	Scan FFs n_s	% Scan	$\mathcal{F}_{OM}$ Cycles	$\mathcal{F}_{OM}$ Faults $\mathcal{F}_{hard}$	$\mathcal{F}_{OM}$ FC (%)	$\mathcal{F}_{TM}$ Cycles	$\mathcal{F}_{TM}$ Faults	$\mathcal{F}_{TM}$ FC (%)	$\mathcal{F}_{OM} \cup \mathcal{F}_{TM}$ Faults	$\mathcal{F}_{OM} \cup \mathcal{F}_{TM}$ FC (%)
s344	15	0	8	53.33	47	13	100.00	64	77	100.00	419	100.00
s967	29	0	6	20.69	57	19	100.00	51	59	100.00	1125	100.00
s991	19	15	3	15.79	84	26	100.00	26	82	100.00	992	100.00
s1196	18	0	8	44.44	126	18	100.00	81	77	98.70	1319	99.92
s1269	37	17	9	24.32	31	4	100.00	131	139	97.21	1482	99.73
s1512	57	2	55	96.49	4827	578	100.00	673	508	100.00	1865	100.00
s3271	116	0	25	21.55	261	14	100.00	260	230	99.13	3500	99.94
s3330	132	0	118	89.39	24189	764	100.00	1430	1067	100.00	3937	100.00
s3384	183	0	94	51.37	6083	273	100.00	1235	851	99.88	4231	99.98
s4863	104	0	71	68.27	2451	233	100.00	1800	644	97.52	5408	99.70
s6669	239	0	18	7.53	59	9	100.00	171	167	99.40	6851	99.99

0.1 to 1200 CPU seconds for *SCORCH* chain assignment, while test generation ranges from 0.2 to 200 CPU seconds. In our case, due to an early prototype implementation, the cost of sequential fault simulation dominates the overall cost of SCORCH.

VI. Conclusions

In this paper, we have presented a new formulation of the partial scan chain assignment problem, subject to meeting a user-prescribed level of fault coverage. The approach as proposed in *SCORCH* is a viable complement to existing partial scan assignment and test generation methods for as long as the cost of sequential fault simulation and combinational test generation remain affordable. A more aggressive assignment strategy, improving the current results, is feasible and will be reported elsewhere.

Acknowledgement. The authors acknowledge Prof. Janek Patel and students at the University of Illinois for the generous assistance with the sequential test generation tool HITEC.

References

[1] S. T. Chakradhar and S. Dey. Resynthesis and Retiming for Optimum Partial Scan. In *ACM/IEEE 31st Design Automation Conference*, pages 87 – 93, June 1994.
[2] Y. Higami, S. Kajihara, and K. Kinoshita. A Partial Scan Algorithm Based on Reduced Scan Shift. In *Third Asian Test Symposium*, pages 277–282, Nov 1994.
[3] S. Dey, M. Potkonjak, and R. Roy. Exploiting Hardware-Sharing in High Level Synthesis for Partial Scan Optimization. In *IEEE International Conference on Computer-Aided Design*, pages 20 – 25, November 1993.
[4] H. Gundlach, B. Koch, and K. Muller-Glaser. On the Selection of a Partial Scan Path with Respect to Target Faults. In *European Design Automation Conference*, 1991.
[5] V. Chickermane and J. Patel. A Fault Oriented Partial Scan Design Approach. In *IEEE International Conference on Computer-Aided Design*, November 1991.
[6] D. Lee. and S. Reddy. On Determining Scan Flip-Flops in Partial Scan Designs. In *IEEE International Conference on Computer-Aided Design*, pages 322–325, November 1990.
[7] V. Chickermane and J. Patel. An Optimization Based Approach to the Partial Scan Problem. In *International Test Conference, IEEE*, pages 377–387, September 1990.
[8] K.T. Cheng and V.D. Agrawal. A Partial Scan Method for Sequential Circuits with Feedback. *IEEE Transactions on Computers*, 39(4):544–548, April 1990.
[9] Rajesh Gupta, Rajiv Gupta and Melvin Breuer. An Efficient Implementation of the BALLAST Partial Scan Architecture. *IEEE Transactions on Computers*, 39(4):538–544, April 1990.
[10] A. Kunzmann and H. Wunderlich. An Analytical Approach to the Partial Scan Problem. *Journal of Electronic Testing: Theory and Application*, 1(2):163–174, 1990.
[11] A. Motohara, T. Ohta, and M. Akino. Critical Flip-Flop Identification Algotithms for Partial Scan Design. In *IFIP Workshop on Design & Test of ASICs*, pages 75–78, 1990.
[12] V. Agrawal, K.-T. Cheng, D. Johnson, and T. Lin. Designing Circuits with Partial Scan. *IEEE Design & Test of Computers*, 5:8–15, April 1988.
[13] S. Bhawmik, K.-T. Cheng, C. Lin, and V. Agrawal. PASCANT: A Partial Scan and Test Generation System. In *IEEE Custom Integrated Circuits Conference*, pages 17.3.1–17.3.4, 1991.
[14] D. Pellkofer. *Fast Fault Simulation in Digital Circuits.* PhD thesis, Dept. of Electrical Engineering, Technical University of Munich, April 1992.
[15] K. Kozminski, (Ed.). OASIS2.0 User's Guide. MCNC, Research Triangle Park, N.C. 27709, 1992.
[16] J. Calhoun and F. Brglez. A Framework and Method for Hierarchical Test Generation. *IEEE Transactions on Computer-Aided Design*, 11(1):45–67, January 1992.
[17] C.-J. Lin, Y. Zorian, and S. Bhawmik. PSBIST: A Partial-Scan Based Built-in Self-Test Scheme. In *International Test Conference, IEEE*, pages 507–516, October 1993.
[18] M. Abramovici, J. Kulikowski, and R. Roy. The Best Flip-Flops To Scan. In *International Test Conference, IEEE*, pages 166–173, October 1991.
[19] T. Chakraborty, V. Agrawal, and M. Bushnell. Design for Testability for Path Delay Faults in Sequential Circuits. In *30th Design Automation Conference, ACM/IEEE*, pages 453–457, June 1993.
[20] K.-T. Cheng. Partial Scan Designs Without a Separate Scan Clock. In *VLSI Test Symposium*, pages 277–282, April 1995.
[21] K. Kim and C. Kime. Partial Scan Using Reverse Direction Empirical Testability. In *International Test Conference, IEEE*, pages 498–506, October 1993.
[22] C. Gloster. ISCAS'89 Addendum Benchmark Set. ACM/SIGDA Benchmarks Electronic Newsletter, F. Brglez (Ed.), June 1993. Available via ftp at ftp.cbl.ncsu.edu or www at http://www.cbl.ncsu.edu/www/; for autoreply e-mail to benchmarks@cbl.ncsu.edu.
[23] T. Niermann and J. Patel. HITEC: A Test Generation Package for Sequential Circuits. In *European Design Automation Conference*, 1991.
[24] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization.* Prentice Hall, Englewood, New Jersey, 1982.
[25] M. Abramovici, P. Parikh, B. Mathew, and D Saab. On Selecting Flip-Flops for Partial Reset. In *IEEE International Test Conference*, pages 1008 – 1012, September 1993.